เปรมนาถ ดูเบ

COM กับการพัฒนาซอฟต์แวร์

COM เป็นสถาปัตยกรรมที่ทำให้ซอฟต์แวร์ต่าง ๆ สามารถติดต่อทำงานร่วมกันได้ โดยใช้มาตรฐานแบบ Binary Object

การพัฒนาซอฟต์แวร์รูปแบบใหม่ที่กำลังได้รับการกล่าวถึงและเป็นที่ยอมรับกันอย่างกว้างขวางในวงการอุตสาหกรรมซอฟต์แวร์ปัจจุบันคือ การพัฒนาซอฟต์แวร์แบบ Component Base (Component Base Development) ซึ่งเป็นการเปลี่ยนวิธีการพัฒนาซอฟต์แวร์ที่ต้องทำการสร้างฟังก์ชันการทำงานทุกอย่างเองหมดและรวมกันอยู่เป็นซอฟต์แวร์ขนาดใหญ่ มาเป็นการพัฒนาชิ้นส่วนย่อย ๆ ที่เรียกว่า Component โดยที่ Component เหล่านี้สามารถนำมาเชื่อมต่อกันสร้างเป็นซอฟต์แวร์ขนาดใหญ่อีกทีหนึ่ง ทั้งนี้ Component ทั้งหลายที่ประกอบเป็นซอฟต์แวร์ของเรานั้นไม่จำเป็นที่จะต้องผลิตเองทั้งหมด เพราะสามารถจะหาซื้อ Component ที่มีคุณสมบัติและความสามารถตรงกับที่เราต้องการมาใช้ได้เลย ซึ่งการผลิตซอฟต์แวร์ Component ขายก็เป็นธุรกิจที่กำลังเติบโตเป็นอย่างมาก ด้วยเหตุนี้จึงมีความพยายามที่จะพัฒนาเครื่องมือต่าง ๆ เพื่อสนับสนุนความต้องการของตลาดที่มีจำนวนและความซับซ้อนเพิ่มมากขึ้น

หลักการในการพัฒนาซอฟต์แวร์ที่เชื่อว่าน่าจะสอดคล้องกับแนวทาง Component Base และก็เป็นที่ยอมรับกันอย่างแพร่หลายในขณะนี้ก็คือการพัฒนาโดยยึดหลัก Object Oriented จุดเด่นของหลักการนี้คือความสามารถที่จะนำ Object ที่มีอยู่แล้วมาใช้ใหม่ ทำให้ขบวนการในการพัฒนาซอฟต์แวร์แต่ละครั้งไม่ต้องเริ่มต้นจากการพัฒนาทุกอย่างใหม่หมด Object สามารถมองดูเหมือนเป็นกล่องดำที่เก็บรายละเอียดเกี่ยวกับ คุณสมบัติ สถานะและวิธีการทำงานไม่ให้ผู้ใช้เห็น ซึ่งวิธีการที่ผู้ใช้จะสามารถติดต่อกับ Object ได้ก็โดยผ่านทางฟังก์ชันที่ Object ได้จัดไว้ให้เท่านั้น ดังนั้นถ้าเรามี Object ที่มีความสามารถตามที่เราต้องการเก็บไว้หรือสามารถหาซื้อได้ การพัฒนาซอฟต์แวร์ตามแนวทาง Object Oriented ก็จะเป็นแค่เพียงการ

นำ Object ต่าง ๆ ที่อยู่มาประกอบเข้าด้วยกัน แล้วอาจจะมีการเขียนเพิ่มเติมในส่วนที่เป็นความต้องการเฉพาะของเราเข้าไป เพียงเท่านั้นเราก็จะได้ซอฟต์แวร์ตามที่ต้องการ (ผมไม่ได้มีเจตนาให้ใครคิดถึงบะหมี่กึ่งสำเร็จรูปหรอกนะครับ)

ดูเหมือนว่าแนวคิดแบบ Object Oriented น่าจะเป็นแนวคิดที่ดีในการพัฒนาซอฟต์แวร์ในแบบ Component Base สำหรับภาษาที่ใช้ในการเขียนโปรแกรมในลักษณะ Object Oriented อย่างเช่น C++, SmallTalk หรือ Java สามารถจะเขียน Class ของ Object ที่ทำงานต่าง ๆ ซึ่งนำมาใช้ใหม่ได้ ทั้งนี้และทั้งนั้นก็เฉพาะกับภาษาเดียวกับมันเท่านั้น นั่นคือเราไม่สามารถนำเอา Class ของ C++ ไปใช้ร่วมกับ Java ได้ (ถึงจะได้แต่ก็ไม่ง่ายนัก) นี่ไม่ต้องพูดถึงภาษาอย่าง Visual Basic ซึ่งไม่ใช่ภาษาแบบ Object Oriented ถ้าเช่นนั้นแล้วทำอย่างไรที่จะให้ซอฟต์แวร์ทั้งหลายที่เขียนขึ้นด้วยภาษาที่ต่างกันสามารถทำงานร่วมกันได้ ทางออกทางหนึ่งก็คือ กำหนดมาตรฐานในการติดต่อแบบ binary มาตรฐานอันหนึ่งที่เกิดขึ้นมาเพื่อแก้ปัญหานี้ก็คือ COM หรือ Component Object Model

มาตรฐานการติดต่อแบบ Binary

ถ้าจะขอให้ท่านผู้อ่านที่เคยได้ยินได้ฟังหรือรู้จักเรื่องเกี่ยวกับ Component Object Model ช่วยยกมือขึ้นหน่อย ครับรอซักครู่ครับขอผมนับก่อน เอาล่ะครับทีนี้ใครที่เคยได้ยินคำว่า ActiveX บ้างช่วยยกมือให้ผมอีกซักทีครับ โอเค ผลเป็นไปตามที่ผมคาดไม่มีผิด คะแนนของ ActiveX กินขาด! ผมจึงอยากขอเวลาสักนิดในการกล่าวถึงที่มาที่ไปของคำเหล่านี้

ก่อนหน้าที่จะมี ActiveX เกิดขึ้นมา เรามีคำว่า OLE หรือ Object Linking and Embedding เป็นเทคโนโลยีมาตรฐานที่ใช้กันอยู่ในระบบปฏิบัติการ Windows เพื่อใช้ทำสิ่งที่เรียกว่า Compound Document ต่อมาความต้องการทางความสามารถอื่น ๆ มีมากขึ้น จึงได้มีการพัฒนา OLE version 2 ขึ้นมา โดยให้มีพื้นฐานอยู่บนมาตรฐานทางเทคโนโลยีตัวใหม่ที่เรียกว่า COM จากนั้นมาก็มีเทคโนโลยีอื่น ๆ อีกที่พัฒนาขึ้นมาบนมาตรฐานของ COM ด้วยเหตุผลทางการตลาด Microsoft จำเป็นจะต้องโปรโมตเทคโนโลยีที่ตนมีอยู่ให้เป็นที่รู้จัก Microsoft เลือกคำว่า OLE เพื่อใช้ในการอ้างถึงเทคโนโลยีทั้งหมดที่อยู่บนมาตรฐานของ COM เมื่อเวลาผ่านไปเข้ามาสู่ยุคที่ Internet ได้รับความนิยมอย่างแพร่หลาย มีความต้องการโปรแกรมขนาดเล็กที่สามารถทำงานบน Web Browser ได้ และต้องส่งผ่านทางเครือข่ายคอมพิวเตอร์ได้อย่างรวดเร็วด้วย นอกจากนั้นการทำงานในลักษณะ Distributed Computing ก็มีความจำเป็นมากขึ้นด้วย ดังนั้นทาง Microsoft ก็ได้เสนอเทคโนโลยีใหม่เพิ่มเติมขึ้นมาเพื่อรองรับความต้องการดังกล่าว แต่ก็ยังคงอยู่บนมาตรฐานของ COM อยู่ แต่จะทำอย่างไรให้เทคโนโลยีนี้เป็นที่รู้จักกันอย่างกว้างขวางและจะได้มีผลทางการตลาดด้วยในขณะเดียวกัน หน่วยงานที่มีความสามารถมากที่สุดหน่วยงานหนึ่งของ Microsoft ก็คือหน่วยงานด้านการตลาดจึงเลือกทำที่ดึงดูดใจมากคำหนึ่งมาใช้เป็นเครื่องหมายการค้าแทนเทคโนโลยีอันนี้ นั่นก็คือ ActiveX ถึงแม้จะสร้างความสับสนให้กับกลุ่มพัฒนาซอฟต์แวร์ในช่วงแรก ๆ อยู่พอสมควร ในเรื่องเกี่ยวกับความหมายที่แท้จริงของมัน แต่ผลในด้านการประชาสัมพันธ์ต้องยอมรับว่าได้ผลอย่างมาก ก็คำว่า ActiveX มันฟังดูแล้วเซ็กซี่ดีออก ว่าไหมครับ

คำจำกัดความของ Component Object Model หรือ COM ค่อนข้างจะกว้างพอสมควร ถ้ามองจากเทคโนโลยีต่าง ๆ ที่ถูกพัฒนาขึ้นมาบนมาตรฐานของมัน แต่โดยหลักแล้ว COM เป็นสถาปัตยกรรมที่ทำให้ซอฟต์แวร์ต่าง ๆ สามารถติดต่อทำงานร่วมกันได้ โดยใช้มาตรฐานแบบ Binary Object คือวิธีการติดต่อกันจะใช้หลักการทาง Object Oriented แต่ Object ที่ว่านี้จะอยู่ในรูปโปรแกรมที่คอมไพล์แล้ว ซึ่งอาจจะเป็น DLL หรือโปรแกรมที่สามารถทำงานได้โดยตัวเองเช่นโปรแกรม EXE ก็ได้ ด้วยวิธีการนี้โปรแกรมที่ติดต่อกันก็ไม่จำเป็นต้องเขียนด้วยภาษาเดียวกันก็ได้ ผลก็คือผู้พัฒนาซอฟต์แวร์สามารถใช้วิธีการมาตรฐานเพียงอย่างเดียวเท่านั้นในการที่จะทำให้ซอฟต์แวร์ของตนติดต่อทำงานร่วมกับซอฟต์แวร์ของผู้อื่นได้ ไม่จำเป็นต้องทราบว่าซอฟต์แวร์ที่เราจะติดต่อด้วยจะเขียนด้วยภาษาอะไร จะเป็น EXE หรือ DLL แม้กระทั่งอยู่ที่ไหนนั่นคือจะอยู่บนเครื่องเดียวกันหรือจะอยู่บนเครื่องอื่นบนเครือข่ายที่เครื่องของเราเชื่อมต่ออยู่ โดยที่ COM จะเป็นตัวจัดการกับรายละเอียดต่าง ๆ เหล่านี้เอง เราสามารถแบ่ง COM Object ออกเป็นสองฝ่าย คือฝ่ายที่เรียก กับฝ่ายที่ถูกเรียก ผมขอเรียกซอฟต์แวร์หรือโปรแกรมไม่ว่าจะเป็น EXE หรือ DLL ฝ่ายที่เรียกว่า COM Client และฝ่ายที่ถูกเรียกหรือให้บริการว่า COM Server

เทคโนโลยีต่าง ๆ ที่พัฒนาขึ้นบนพื้นฐานของ COM ที่สำคัญได้แก่

- OLE
- Automation
- Control

OLE หรือ Object Linking and Embedding เป็นเทคโนโลยีที่ทำให้เกิดสิ่งที่เรียกว่า Compound Document ซึ่งสามารถแบ่งออกเป็นสองชนิดตามชื่อก็คือ Object Linking และ Object Embedding ตัวอย่างของ Compound Document ที่เห็นอยู่โดยทั่วไปก็ได้แก่ การที่เรา Insert Object ใน Microsoft Word ไม่ว่าจะเป็นรูปจากโปรแกรม Paint หรือตารางจาก Excel เป็นต้น

Automation คือความสามารถที่ COM Client สามารถเรียกใช้บริการจาก Object ของ COM Server เพื่อทำงานบางอย่างที่เป็นความสามารถเฉพาะของตัว Server ซึ่งจะช่วยให้ Client ไม่จำเป็นต้องพัฒนาฟังก์ชันดังกล่าวเพิ่มเติมเข้ามาในตัวเอง ทำให้ลดเวลาในการพัฒนาซอฟต์แวร์ไปได้เป็นอย่างมาก

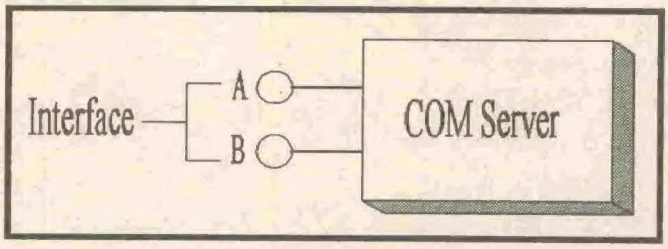
Control ที่ทราบกันดีในชื่อ ActiveX Control มีลักษณะเป็นโปรแกรมขนาดเล็กที่สามารถนำมาใส่ในไฟล์ HTML เพื่อใช้ทำงานบน Web Browser นอกจากนั้น Control ยังเป็นเทคโนโลยีที่โปรแกรมเมอร์ภาษา Visual Basic รู้จักกันดีในนาม OCX ซึ่งมีความสำคัญมากในการพัฒนาโปรแกรมบน Windows ในปัจจุบัน

โครงสร้างพื้นฐานของ COM

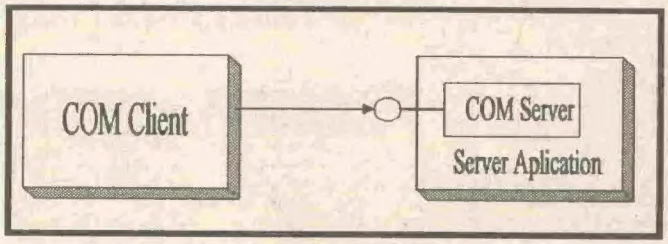
COM มีลักษณะเป็น Object ที่เก็บรายละเอียดในการทำงานและข้อมูลต่าง ๆ ไว้ภายในเหมือนกับกล่องดำที่เราสามารถมองเห็นได้เฉพาะรูปร่างภายนอกเท่านั้น แต่อย่านำไปเปรียบเทียบกับ Object ของ Class ที่ได้จากภาษา Object Oriented ทั้งหลาย ทั้งในแง่ของการสร้างและการใช้งาน จะเหมือนกันก็เพียงแต่แนวความคิดเท่านั้น นั่นคือทฤษฎีทางด้าน Object Oriented ที่เกี่ยวข้องเช่น Encapsulation, Polymorphism และ Inheritance ซึ่งถูกนำมาใช้ในการทำงานร่วมกับ COM ด้วยนั่นเอง ดังนั้นการที่จะทำความเข้าใจการทำงานของ COM ให้ดีแล้วนั้นจำเป็นจะต้องมีพื้นฐานความรู้ด้าน Object Oriented อยู่บ้าง

โดยมากแล้ว Object ของ COM จะแสดงโดยใช้รูปสี่เหลี่ยมและมีรูปกลมเล็ก ๆ เชื่อมอยู่ ดังที่แสดงในรูปที่ 1

COM Server ใช้ส่วนที่เรียกว่า Interface ในการติดต่อสื่อสารกับโลกภายนอกและเป็นเพียงหนทางเดียวเท่านั้นที่ Object ของ COM จะสามารถติดต่อกับ COM Client ซึ่ง Interface จะแทนด้วยสัญลักษณ์รูปกลมที่เชื่อมออกมาจากตัวกล่อง COM Server หนึ่ง ๆ จะมี Interface กี่อันก็ได้ โดยที่ Interface แต่ละอันจะประกอบไปด้วยกลุ่มของฟังก์ชันต่าง ๆ ที่สัมพันธ์กันและมีไว้เพื่อให้บริการแก่ COM Client ดูรูปที่ 2 ประกอบ



รูปที่ 1
แสดง Object ของ COM



รูปที่ 2
แสดงความสัมพันธ์ระหว่าง COM Server กับ COM Client

ถ้าใช้หลักการทางด้าน Object Oriented มาอธิบายจะพบว่า เราสามารถจะเปรียบเทียบ Interface ได้กับ pure abstract base class ที่มีเฉพาะฟังก์ชัน เราจะลองสมมติวิธีการทำงานของ Interface โดยใช้ภาษา C++ ซึ่งสามารถจะแสดง Interface ที่ชื่อ IFoo ได้ดังนี้

```
class IFoo
{
    virtual void Func1(void) = 0;
    virtual void Func2(int count) = 0;
};
```

ใน C++ นั้น pure abstract class ไม่สามารถจะสร้าง Instance หรือ Object ของ Class ได้และจะไม่มีส่วนของการทำงานของฟังก์ชันอยู่ด้วย ซึ่งจะเห็นได้ว่าฟังก์ชันจะถูกกำหนดให้เป็น Virtual วิธีการที่จะใช้ abstract class ได้ก็โดยการสืบทอด (Inherit) class ใหม่ต่อลงมา แล้วกำหนดรายละเอียดในส่วนของการทำงานของแต่ละฟังก์ชันลงไป ตัวอย่างเช่น

```
class CFoo : public IFoo
{
    void Func1(void) { /* ... */ }
    void Func2(int nCount) { /* ... */ }
};
```

เวลาที่มีการกำหนดให้ฟังก์ชันเป็น Virtual Compiler จะสร้าง virtual funtion table หรือที่เรียกกันว่า vtable ซึ่งเป็นตารางของ pointer ที่ชี้ไปที่ฟังก์ชัน เพื่อใช้หาตำแหน่งที่อยู่และการอ้างอิงของฟังก์ชันในสายการสืบทอดของ class ที่ต่อลงมาจาก abstract base class ซึ่งในที่นี้คือฟังก์ชัน Func1 และ Func2 ใน CFoo ที่มีส่วนของการทำงานของฟังก์ชันชื่อเดียวกันนั้นใน base class

เมื่อเราต้องการจะติดต่อกับ Interface ก็ทำได้ในลักษณะนี้

```
IFoo *pFoo = SomeFunction( ); // ฟังก์ชันส่ง pointer ของ class CFoo ให้กับ pointer ของ IFoo
```

สิ่งที่เห็นข้างบนนี้นั้นเกิดขึ้นได้โดย polymorphism แต่บางท่านอาจจะสงสัยว่าทำไมถึงไม่สร้าง Instance ของ class CFoo โดยตรงแทนที่จะทำผ่านฟังก์ชัน SomeFunction คือโดย

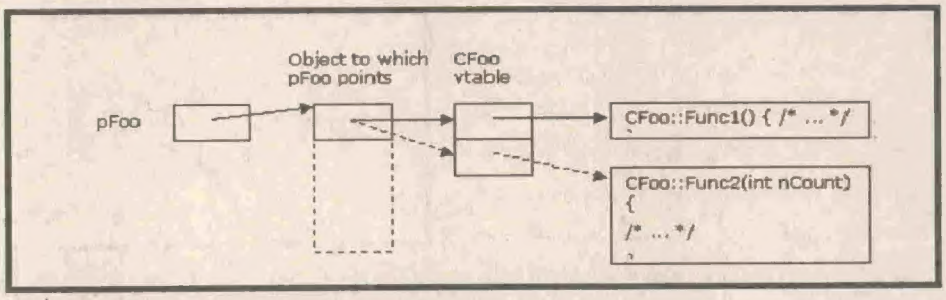
```
IFoo *pFoo = new CFoo( );
```

คำตอบก็คือจริง ๆ แล้ว COM Client จะไม่มีโอกาสได้ติดต่อกับ CFoo โดยตรงได้เลย การสร้าง Instance ของ CFoo จะเป็นหน้าที่ของ class ผู้ช่วยที่เรียกว่า factory class เพื่อเหตุผลในการปิดบังหรือ encapculate รายละเอียดในการทำงานจาก COM Client อย่างที่เคยกล่าวไว้แล้วเราจะสามารถติดต่อกับ COM Object ก็โดยผ่านทาง Interface เท่านั้น แล้วใช้ Interface pointer ที่ได้มาใช้เรียกฟังก์ชันต่าง ๆ อีกที

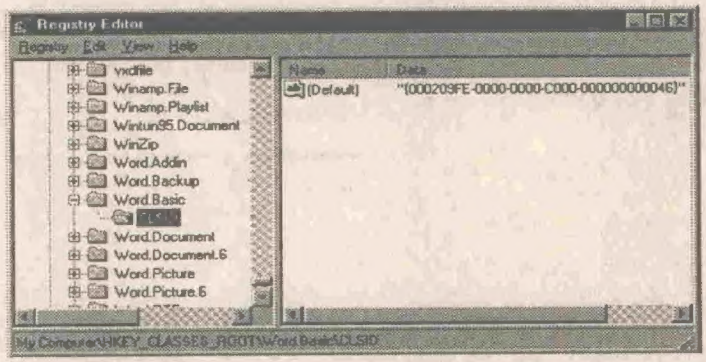
```
pFoo->Func1( );
pFoo->Func2( 5 );
```

ซึ่งสิ่งที่เกิดขึ้นเบื้องหลังของการเรียกฟังก์ชันทั้งสองนั้นจะเป็นไปตามรูปที่ 3

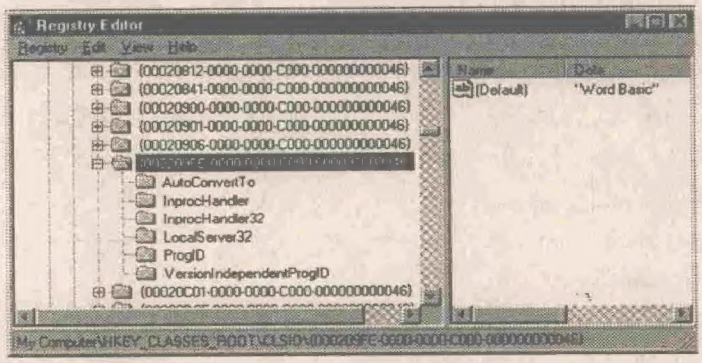
มาถึงประเด็นสำคัญที่ผมกล่าวไว้ตั้งแต่แรกที่ว่า COM ใช้วิธีการติดต่อโดยมาตรฐานแบบ Binary ซึ่งก็คือการติดต่อกันโดยใช้ pointer ของ Interface ดังที่เห็น ภาษาใดก็ตามที่สามารถเรียกฟังก์ชันผ่าน pointer ได้ (C, C++, Java, Smalltalk รวมทั้ง BASIC ด้วย) ก็สามารถจะใช้มาตรฐานนี้ได้



รูปที่ 3 แสดงวิธีการเรียกฟังก์ชันผ่าน Interface pointer



รูปที่ 4 แสดง CLSID ของ Microsoft Word ใน Registry



รูปที่ 5 แสดง CLSID และข้อมูลอื่น ๆ ที่ COM ใช้ในการทำงาน

COM in Action !

เวลาที่ Install ซอฟต์แวร์หรือโปรแกรมใด ๆ ก็ตามที่เป็น COM Server ด้วย มันจะ register ตัวเองไว้ใน Registry ของ Windows คุณสามารถจะดูรายละเอียดของ register ของ COM Server ได้โดยใช้โปรแกรม REGEDIT ลองใช้ Run เมนูจาก taskbar ใส่คำว่า regedit เพื่อเรียกให้โปรแกรมทำงาน

สมมติว่าคุณมี Microsoft Word ลงอยู่ในเครื่อง ในโปรแกรม REGEDIT เข้าไปที่ HKEY_CLASSES_ROOT\Word.Basic (ปกติแล้วสมาชิกแต่ละตัวในฝั่งซ้ายมือใน Registry จะเรียกว่า Key ส่วนในฝั่งขวาเรียกว่า data item) จะเห็นว่ามี Key ชื่อ CLSID ซึ่ง data item ของมันจะมีค่า {000209FE-0000-0000-C000-000000000046} ดูรูปที่ 4 ประกอบ

ค่าที่เห็นเป็นค่าที่เรียกว่า GUID หรือ Globally Unique Identifiers ซึ่งเป็นค่าเฉพาะที่ถูกสร้างขึ้นโดย COM Library ของ Windows ที่มีหน้าที่รับผิดชอบการทำงานที่เกี่ยวข้องกับ COM ซึ่ง Library นี้ใน Windows 3.x จะอยู่ใน COMOBJ.DLL และ OLE32.DLL สำหรับ Windows 95 และ NT ขนาดของเนื้อที่ที่ใช้สำหรับเก็บ GUID นั้นเท่ากับ 16 byte ในที่นี้ใช้เพื่อกำหนดค่าให้กับ Object ของ COM Server แต่ละตัวที่เรียกว่า CLSID หรือ Class Identifier GUID ดังนั้น COM Server แต่ละตัวจะมีค่า CLSID เฉพาะของมันที่ไม่ซ้ำกันกับคนอื่นและจะมีค่าเดียวกันทุกเครื่อง ดังนั้นเครื่องของคุณกับของผมจึงมีค่านี้ตรงกัน ระบบจะใช้ CLSID แทนชื่อในการค้นหาว่ามี COM Server ดังกล่าวอยู่ที่ไหนและในไฟล์ชื่ออะไร เหตุที่ไม่ใช้ชื่อก็เพราะอาจเป็นไปได้ที่จะมีการตั้งชื่อซ้ำกันก็ได้ ซึ่งจะทำให้เกิดปัญหาเวลาเรียกใช้ แต่ข้อมูลเท่าที่เห็นข้างต้นนั้นยังไม่เพียงพอ เพราะเวลาที่ COM Client ต้องการติดต่อกับ Server ระบบยังจำเป็นต้องทราบที่อยู่และชื่อไฟล์ที่แท้จริงที่มี COM Server นั้นอยู่ด้วย

รูปที่ 5 นี้อยู่ที่ Key HKEY_CLASSES_ROOT\CLSID\ ซึ่งเป็นตำแหน่งของ CLSID ของ Word.Basic ถ้าคุณเข้าไปที่ Key ย่อย LocalServer32 จะเห็นว่า data item เป็นตำแหน่งที่อยู่และชื่อไฟล์ที่เป็น COM Server สำหรับเครื่องของผมจะเป็น D:\MSOFFICE\WINWORD\WINWORD.EXE ซึ่งก็คือไดเรคทอรีที่เก็บโปรแกรม Microsoft Word ไว้นั่นเอง ลองดูต่อไปอีกสักนิดไปที่ Key ย่อย ProgID จะมี data item เป็น "word.basic"

ถึงตอนนี้เราคงพอสรุปกันได้แล้วว่าเกิดอะไรขึ้นเวลาที่ COM Client ต้องการติดต่อกับ COM Server นั่นคือ

1. COM Client เรียกขอติดต่อกับ COM Server โดยผ่านทาง CLSID

2. COM Library ใช้ CLSID ไปค้นหา Key ใน Registry

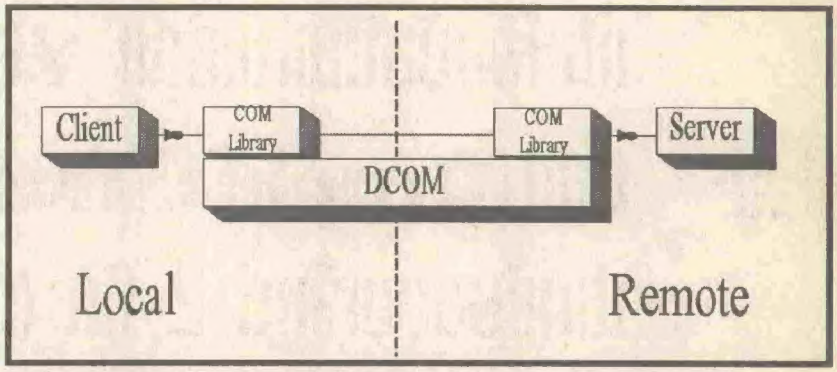
3. COM Library เรียกให้ COM Server ทำงาน แล้วส่ง Interface pointer ให้ COM Client

4. เริ่มต้นการทำงานร่วมกันของ COM Client กับ COM Server โดยผ่าน Interface pointer ของ COM Server

DCOM : จะอยู่ที่ไหนไม่สำคัญ

จากที่ผมกล่าวไปแล้วข้างต้น COM Client สามารถจะติดต่อกับ COM Server ได้โดยไม่จำเป็นต้องทราบว่าตัว Server จะเป็น DLL หรือ EXE ไม่ต้องทราบว่าอยู่ที่ไหนในเครื่อง นอกจากนั้นด้วยความสามารถของ DCOM เราไม่จำเป็นต้องทราบด้วยว่าอยู่ที่เครื่องไหนบนเครือข่ายคอมพิวเตอร์ที่เครื่องของเราเชื่อมต่ออยู่

DCOM หรือ Distributed COM เป็นเทคโนโลยีที่เพิ่มเติมขึ้นจาก COM เพื่อให้ COM Object ทั้งหลายสามารถติดต่อกันผ่านระบบเครือข่ายคอมพิวเตอร์ DCOM ทำให้การพัฒนา COM Object นั้นง่ายขึ้นเพราะไม่ต้องสนใจว่าการใช้งานจริงนั้นจะเกิดขึ้นบนเครื่องเดียวกันหรือต่างเครื่องกัน COM และ DCOM เป็นผู้จัดการกับความซับซ้อนเหล่านี้ให้เราเอง ดูแผนภาพการทำงานในรูปที่ 6



รูปที่ 6 แสดงการทำงานของ COM ร่วมกับ DCOM

นอกจากระบบปฏิบัติการในตระกูล Windows ด้วยกันแล้ว ขณะนี้ได้มีการพัฒนาให้สามารถใช้ COM/DCOM ข้ามไปบนระบบปฏิบัติการอื่น ๆ เช่น Macintosh และ UNIX ได้อีกด้วย

ปัจจุบันและอนาคต

Component Base Development เป็นกระแสที่มีความสำคัญมากในวงการอุตสาหกรรมซอฟต์แวร์ในปัจจุบัน เนื่องมาจากเงื่อนไขหลาย ๆ อย่างเช่น ความต้องการซอฟต์แวร์ลักษณะ Distributed Computing ที่มีสูงขึ้น ซึ่งจำเป็นจะต้องลดปัญหาเรื่องความยุ่งยากในการพัฒนา การดูแลรักษา และความปลอดภัยของระบบ ความต้องการที่จะให้ซอฟต์แวร์ที่มีอยู่แล้วสามารถติดต่อใช้งานร่วมกันได้โดยไม่ขึ้นกับบริษัทที่ผลิต ความต้องการวิธีแก้ปัญหาระหว่างภาษาโปรแกรมที่แตกต่างกัน เงื่อนไขทางด้านเวลาที่ใช้ในการพัฒนาซอฟต์แวร์ การขยายตัวของ Internet ซึ่งทำให้มีความต้องการเทคโนโลยีเฉพาะที่สามารถทำงานบน Web Browser ได้ เช่น Java หรือ ActiveX Control ความสามารถในการนำกลับมาใช้ใหม่ (Reusability) และปัญหาเกี่ยวกับการติดต่อกันระหว่างซอฟต์แวร์ข้าม platform เหล่านี้คือปัญหาต่าง ๆ ที่ COM และเทคโนโลยีที่พัฒนาขึ้นมาบนพื้นฐานของ COM พยายามจะแก้ไข แต่ COM ก็ไม่ใช่เทคโนโลยีหนึ่งเดียวที่มีอยู่ ยังมี Corba ซึ่งมีบริษัทยักษ์ใหญ่จำนวนมากให้การสนับสนุนและเป็นคู่แข่งที่สำคัญของ COM ในปัจจุบัน

ในอนาคตอันใกล้นี้ Microsoft จะนำ COM เวอร์ชันใหม่คือ COM+ ออกมาใช้งาน ซึ่งผมจะขอไม่กล่าวถึงรายละเอียดตรงนี้ แต่ผมอยากจะบอกว่า ไม่ว่าการแข่งขันทางด้านการตลาดจะเป็นเช่นไร สำหรับผู้ใช้เทคโนโลยีแล้วความก้าวหน้าและพัฒนาการเหล่านี้ล้วนแล้วแต่จะนำประโยชน์และความสะดวกสบายมาให้ทั้งสิ้น แต่สำหรับผู้ที่อยู่ในวงการไอที ผู้ที่ทำหน้าที่พัฒนาและผู้ที่กำหนดนโยบายไม่ว่าจะในระดับใดก็ตาม จำเป็นจะต้องตระหนักถึงแนวโน้มที่กำลังจะเปลี่ยนแปลงวิธีการคิดและการปฏิบัติต่อการพัฒนาเทคโนโลยีทางซอฟต์แวร์ในปัจจุบัน ใครก็ตามที่สามารถนำเอาความก้าวหน้าทางเทคโนโลยีเหล่านี้มาประยุกต์ใช้ให้เป็นประโยชน์แก่ตนได้ จะพบว่าตนเองได้ก้าวพ้นความยุ่งยากอันมากมายที่เคยต้องเสียเวลาเป็นอย่างมากในอดีตมาแล้ว

เอกสารอ้างอิง :

1. Inside OLE second edition โดย Kraig Brockschmidt สำนักพิมพ์ Microsoft press, 1995

2. Microsoft Com Technology, http://www.microsoft.com/cominfo/default.asp

3. Microsoft MSDN, http://msdn.microsoft.com/developer/default.htm